USER GUIDE FOR TWO MICROPHONE DIRECTION OF ARRIVAL ESTIMATION ON ANDROID SMARTPHONE FOR HEARING AID APPLICATIONS

Abdullah Kucuk, Anshuman Ganguly, Yiya Hao,
Dr. Issa M.S. Panahi

STATISTICAL SIGNAL PROCESSING LABORATORY (SSPRL)
UNIVERSITY OF TEXAS AT DALLAS

MAY 2019

This work was supported by the National Institute of the Deafness and Other Communication Disorders (NIDCD) of the National Institutes of Health (NIH) under the award number 5R01DC015430-03. The content is solely the responsibility of the authors and does not necessarily represent the official views of the NIH.

Table of Contents

INTRODUCTION

The 'Two Microphone Direction of Arrival (DOA) Estimation' app is designed for estimation of the speaker direction for hearing aid users in real time direction on a graphical user interface (GUI). The application is trained to perform in noisy conditions as well. The contents of this user guide gives you the steps to implement the 'Two Microphone Direction of Arrival (DOA) Estimation' algorithms on Android devices (that have two microphones) and the steps to be followed after installing the app on the smartphone. This app will be an open source and portable research platform for hearing improvement studies.

This user guide covers the software tools required for implementing the algorithm, how to run C codes on Android devices and usage of other tools that are quite helpful in creating audio apps for audio playback in real time.

The MATLAB and C codes used for the 'Two Microphone Direction of Arrival (DOA) Estimation' algorithm are made available publicly on the following website: http://www.utdallas.edu/ssprl/hearing-aid-project/. The codes can be accessed and used with proper consent of the author for further improvements in research activities related to hearing aids.

The screenshot of the first look of our app is as shown in Figure 1.

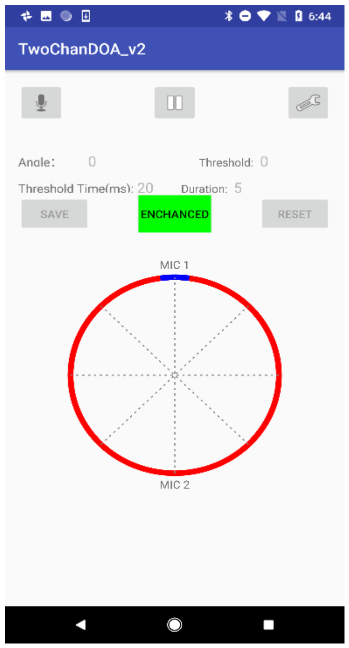


Figure 1

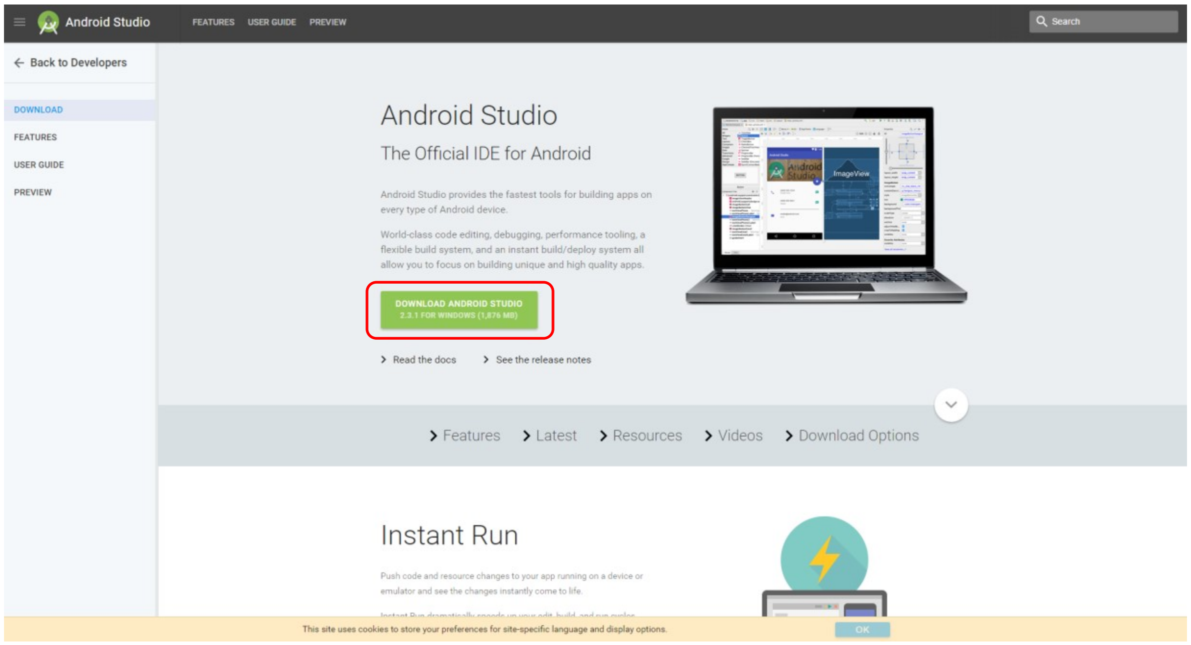
1. SOFTWARE TOOLS

Android is an open-source operating system developed by Google for mobile phones and tablets. The Android apps are usually coded in Java. In this section, it is shown how to set up the Android Studio IDE (Integrated Development Environment) for developing Android apps.

Android Studio IDE requires either Windows Operating System or Apple Operating System. Android Studio IDE can directly build and upload source codes into Android smartphone or generate a APK file which can be downloaded and installed on the Android smartphone.

To download the latest version of Android Studio

1. Open the Android Studio website to download. (https://developer.android.com/studio/index.html)
2. Click Download Android Studio Button (Figure 2).
3. Install the execution file after download.



Android Studio

The Official IDE for Android

Android Studio provides the fastest tools for building apps on every type of Android device.

World-class code editing, debugging, performance tooling, a flexible build system, and an instant build/deploy system all allow you to focus on building unique and high quality apps.

Instant Run

Push code and resource changes to your app running on a device or emulator and see the changes instantly come to life.

This site uses cookies to store your preferences for site-specific language and display options.

Figure 2

2. BUILD AN ANDROID APP

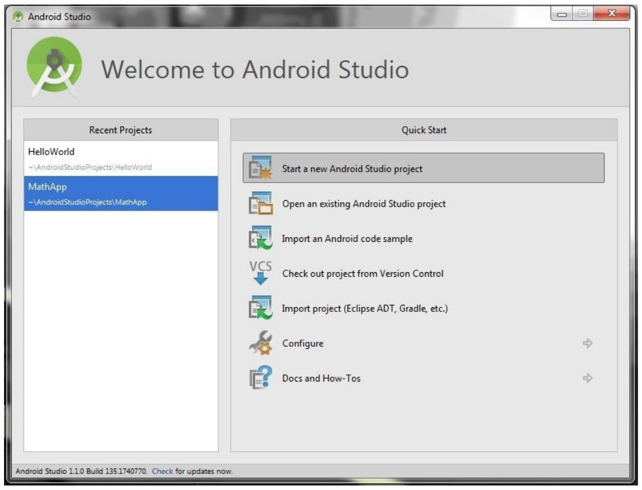
2.1 Programming Language

For creating Android apps, Java is used to create the required shell. The Java Development Kit (JDK) needs to be firstly installed on your computer. This link contains the latest version of JDK: http://www.oracle.com/technetwork/java/javase/downloads/index.html

2.2 Creating Android Apps

After installations of Android Studio and JDK completed, android apps can be created using Android Studio.

1. Open Android Studio.
2. Under the "Quick Start" menu, select "Start a new Android Studio project." (Figure 3)
3. On the "Create New Project" window that opens, name your project "HelloWorld".(Figure 4)
4. If you choose to, set the company name as desired*.
5. Note where the project file location is and change it if desired.
6. Click "Next."
7. Make sure on that "Phone and Tablet" is the only box that is checked. (Figure 4)
8. If you are planning to test the app on your phone, make sure the minimum SDK is below your phone's operating system level.
9. Click "Next."
10. Select "Blank Activity." (Figure 4)
11. Click "Next."
12. Leave all of the Activity name fields as they are. (Figure 4)
13. Click "Finish."



Android Studio

Welcome to Android Studio

Recent Projects

HelloWorld
~\AndroidStudioProjects\HelloWorld

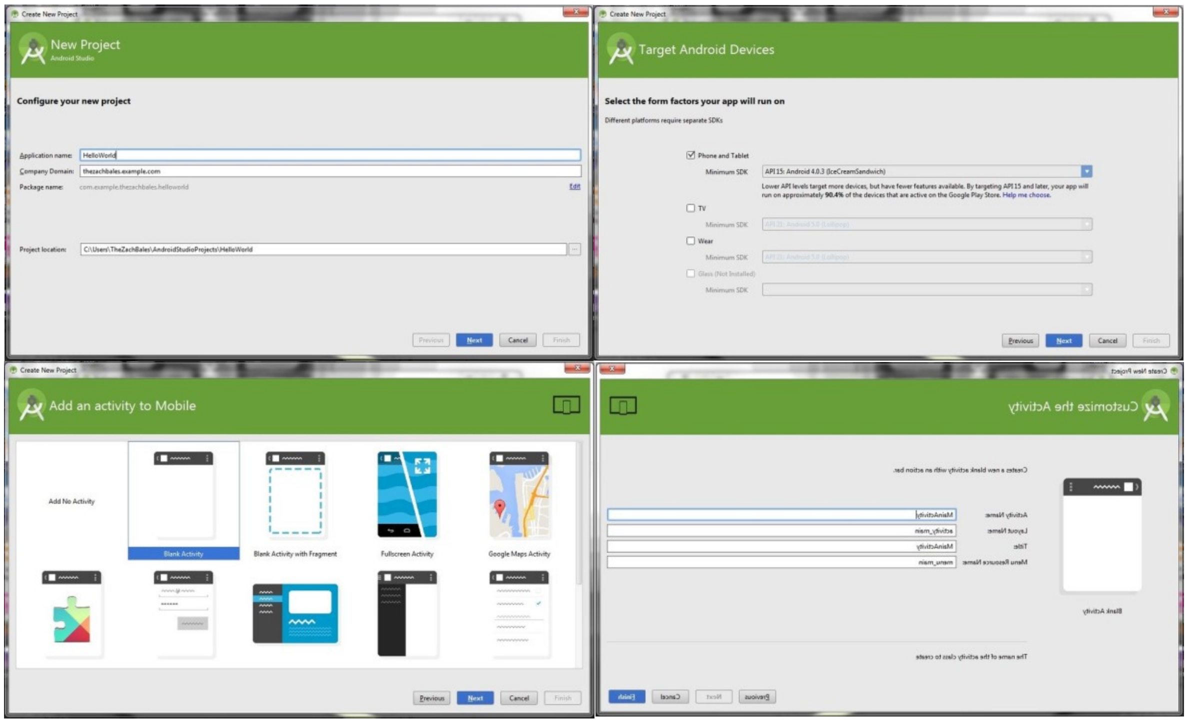
MathApp
~\AndroidStudioProjects\MathApp

Quick Start

- Start a new Android Studio project
- Open an existing Android Studio project
- Import an Android code sample
- Check out project from Version Control
- Import project (Eclipse ADT, Gradle, etc.)
- Configure
- Docs and How-Tos

Android Studio 1.1.0 Build 135.1740770. Check for updates now.

Figure 3



Create New Project

New Project
Android Studio

Configure your new project

Application name:	HelloWorld
Company Domain:	thezachbales.example.com
Package name:	com.example.thezachbales.helloworld
Project location:	C:\Users\TheZachBales\AndroidStudioProjects\HelloWorld

Create New Project

Target Android Devices

Select the form factors your app will run on

Different platforms require separate SDKs

[✓] Phone and Tablet
Minimum SDK: API 15: Android 4.0.3 (IceCreamSandwich)
Lower API levels target more devices, but have fewer features available. By targeting API 15 and later, your app will run on approximately **90.4%** of the devices that are active on the Google Play Store. Help me choose.

[ ] TV
Minimum SDK: API 21: Android 5.0 (Lollipop)

[ ] Wear
Minimum SDK: API 21: Android 5.0 (Lollipop)

[ ] Glass (Not Installed)
Minimum SDK:

Create New Project

Add an activity to Mobile

Add No Activity · Blank Activity · Blank Activity with Fragment · Fullscreen Activity · Google Maps Activity

Create New Project

Customize the Activity

Creates a new blank activity with an action bar.

Activity Name	MainActivity
Layout Name	activity_main
Title	MainActivity
Menu Resource Name	menu_main

Blank Activity

The name of the activity class to create

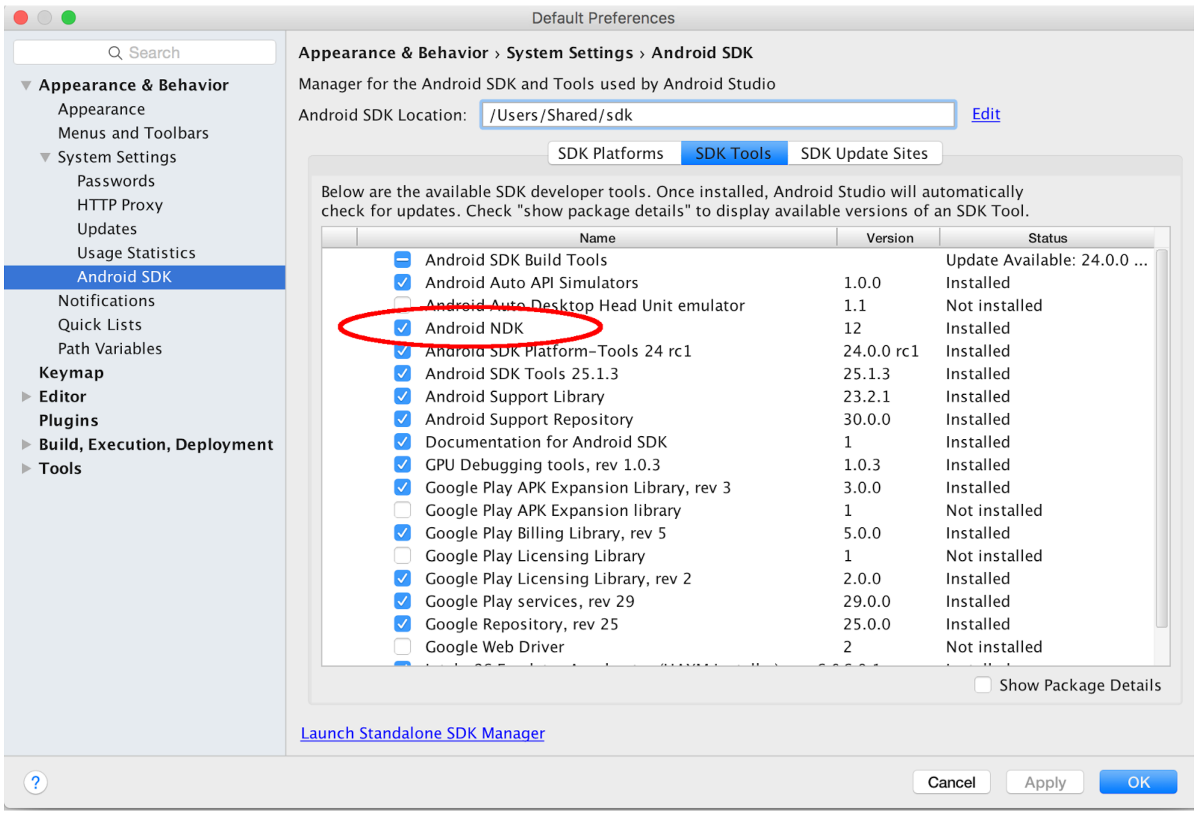
Figure 4

2.3 Adding C File

Android apps are typically written in Java, with its elegant object-oriented design. However, at times, you need to overcome the limitations of Java, such as memory management and performance, by programming directly into Android native interface. Android provides Native Development Kit (NDK) to support native development in C/C++, besides the Android Software Development Kit (Android SDK) which supports Java.

2.3.1 Installing the Native Development Kit (NDK)

1. Menu "Tools" > "Android" > "SDK Manager" (Figure 5)
2. Select tab "SDK Tools"
3. Check "Android NDK"[or "NDK"] if it is not checked
4. Sync and re-build the project.



Appearance & Behavior › System Settings › Android SDK

Manager for the Android SDK and Tools used by Android Studio

Android SDK Location: /Users/Shared/sdk

Below are the available SDK developer tools. Once installed, Android Studio will automatically check for updates. Check "show package details" to display available versions of an SDK Tool.

	Name	Version	Status
−	Android SDK Build Tools		Update Available: 24.0.0 ...
✓	Android Auto API Simulators	1.0.0	Installed
	Android Auto Desktop Head Unit emulator	1.1	Not installed
✓	Android NDK	12	Installed
✓	Android SDK Platform-Tools 24 rc1	24.0.0 rc1	Installed
✓	Android SDK Tools 25.1.3	25.1.3	Installed
✓	Android Support Library	23.2.1	Installed
✓	Android Support Repository	30.0.0	Installed
✓	Documentation for Android SDK	1	Installed
✓	GPU Debugging tools, rev 1.0.3	1.0.3	Installed
✓	Google Play APK Expansion Library, rev 3	3.0.0	Installed
	Google Play APK Expansion library	1	Not installed
✓	Google Play Billing Library, rev 5	5.0.0	Installed
	Google Play Licensing Library	1	Not installed
✓	Google Play Licensing Library, rev 2	2.0.0	Installed
✓	Google Play services, rev 29	29.0.0	Installed
✓	Google Repository, rev 25	25.0.0	Installed
	Google Web Driver	2	Not installed

Figure 5

2.3.2 Writing a Hello-World Android NDK Program

Creating a new project with support for native code is similar to creating any other Android Studio project, but there are a few additional steps:

1. In the Configure your new project section of the wizard, check the Include C++ Support checkbox.
2. Click Next.
3. Complete all other fields and the next few sections of the wizard as normal.
4. In the Customize C++ Support section of the wizard, you can customize your project with the following options:
   - C++ Standard: use the drop-down list to select which standardization of C++ you want to use. Selecting Toolchain Default uses the default CMake setting.
   - Exceptions Support: check this box if you want to enable support for C++ exception handling. If enabled, Android Studio adds the -fexceptionsflag to cppFlags in your module-level build.gradle file, which Gradle passes to CMake.
   - Runtime Type Information Support: check this box if you want support for RTTI. If enabled, Android Studio adds the -frtti flag to cppFlags in your module-level build.gradle file, which Gradle passes to CMake.
5. Click Finish.

After Android Studio finishes creating your new project, open the Project pane from the left side of the IDE and select the Android view. As shown in figure 6, Android Studio adds the cpp and External Build Files groups:

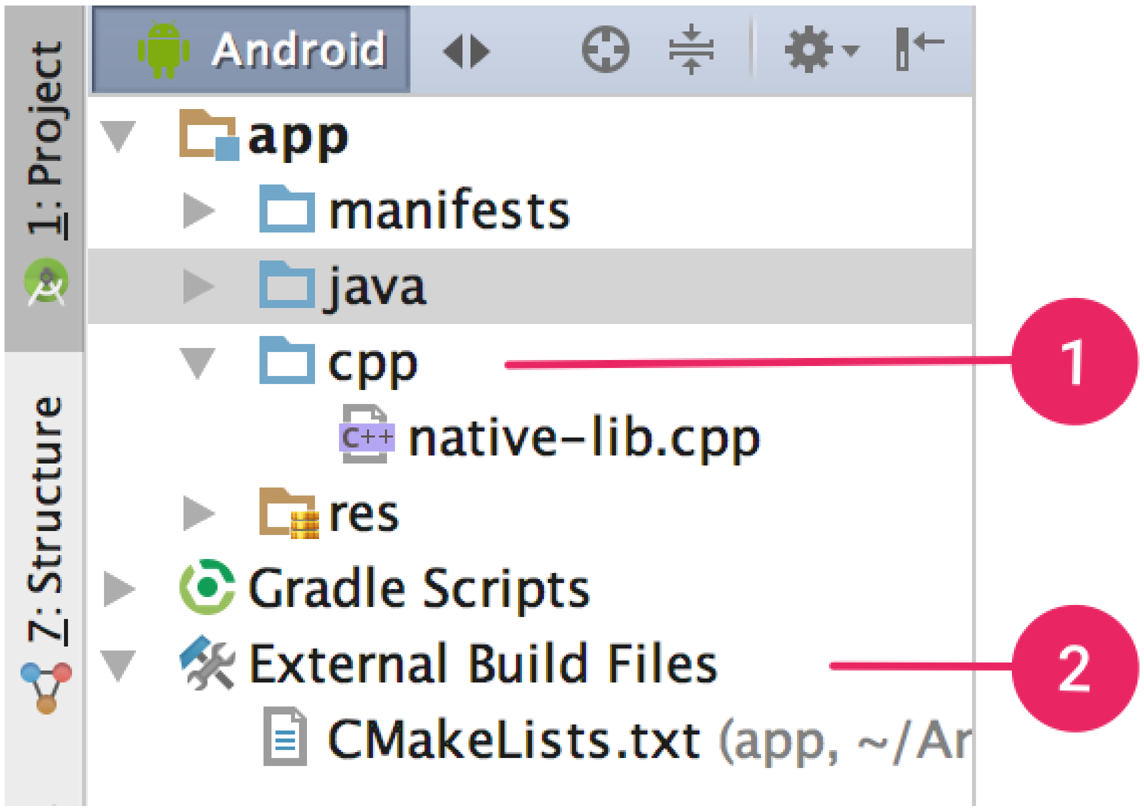


Figure 6

1. The cpp group is where you can find all the native source files, headers, and prebuilt libraries that are a part of your project. For new projects, Android Studio creates a sample C++ source file, native-lib.cpp, and places it in the src/main/cpp/ directory of your app module. This sample code provides a simple C++ function, stringFromJNI(), that returns the string “Hello from C++”.

2. The External Build Files group is where you can find build scripts for CMake or ndk-build. Similar to how build.gradle files tell Gradle how to build your app, CMake and ndk-build require a build script to know how to build your native library. For new projects, Android Studio creates a CMake build script, CMakeLists.txt, and places it in your module’s root directory.

3. STEREO MICROPHONE ACCESS

Accessing two microphones for Android devices is done by rooting smartphone in previous version. This problem is solved for with this version using proposed framework for audio signal processing [1]. One of these libraries and classes is AudioRecord class which administer the audio sources for recording audio from built-in microphones of the platform [2]. We use Android libraries for accessing two mics at the same. A few parameters have critical importance for signal capture. These are explained as follows.

- *Audio Source* = AudioRecord class allows us to select an audio source of recording. Although there are many options to choose a source, it is recommended that MIC and CAMCORDER. MIC denotes for the bottom microphone of the smartphone and CAMCORDER is used for microphone audio source which is near to the camera.
- *Sampling Frequency* = It is used for determining the sampling rate of recording. According to the developer of Android operating system, 44.1 kHz or 48 kHz is suggested as a sampling frequency for modern Android smartphones. These sampling rates provide high quality and minimum audio latency [3].
- *Channel Configuration* = Number of audio channel is defined via channel configuration parameter. Stereo channel can be selected at max. However, Android doesn't guarantee two-channel recording for all Android phones. Google Pixel 2 XL, Google Pixel, and Samsung Galaxy S7 are tested, and it has seen that two-channel recording is possible for these phones.
- *Audio Format* = The audio data is represented by ENCODING_PCM_16BIT. Each sample is denoted by 16 bit, and Pulse Code Modulation is used for the data.
- *Buffer Size* = The size of recorder buffer is crucial. Since we don`t desire to lose any data when capturing signal. Hence, getminbufsize() function is used for determining buffer size. In order to avoid any data losing while signal capturing, function getminbufsize() has been used for determining the minimal buffer size. The return value from the function above is assigned to Buffer Size.

We have used three different smartphones which is shown Figure 7 for real time Direction of Arrival Estimation applications.

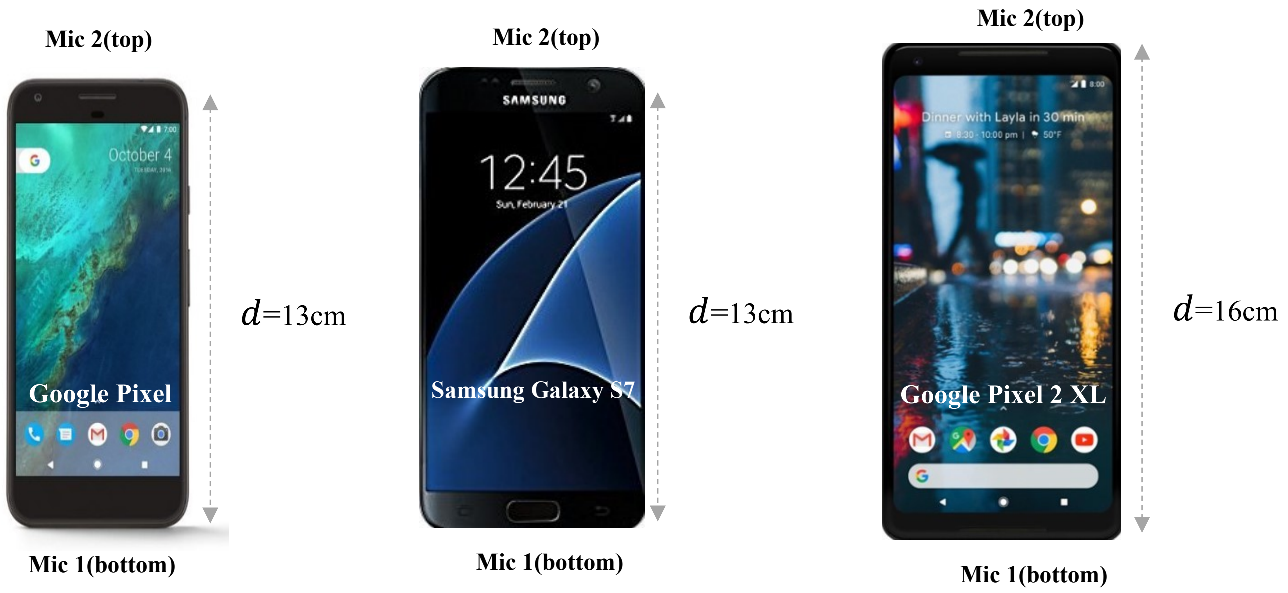

Mic 2(top)
Google Pixel
d=13cm
Mic 1(bottom)
Mic 2(top)
SAMSUNG
12:45
Samsung Galaxy S7
d=13cm
Mic 1(bottom)
Mic 2(top)
Dinner with Layla in 30 min
Google Pixel 2 XL
d=16cm
Mic 1(bottom)

Figure 7

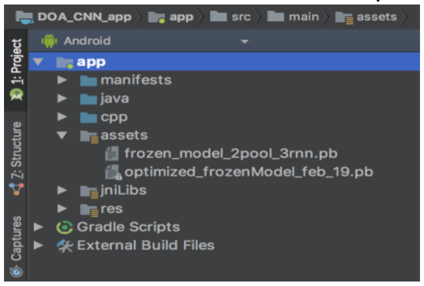
4. DEPLOYING TENSORFLOW MODEL TO ANDROID

This section explains how to deploy pre-trained model to Android platform which is required for DOA application version 3. We have utilized TensorFlow Mobile for inferring DOA angle using pre-trained Deep Learning TensorFlow model. Before deploying, your model should be trained. We have explained model training in our paper for DOA estimation[ref]. After training model should be frozen. Following steps shoul be followed to deploy TensorFlow model to Android.

- TensorFlow libraries should be imported (in app's build.gradle)

```
implementation files('libs/libandroid_tensorflow_inference_java.jar')
```

- The frozen model should be put into the "assets" file in Android project directory.

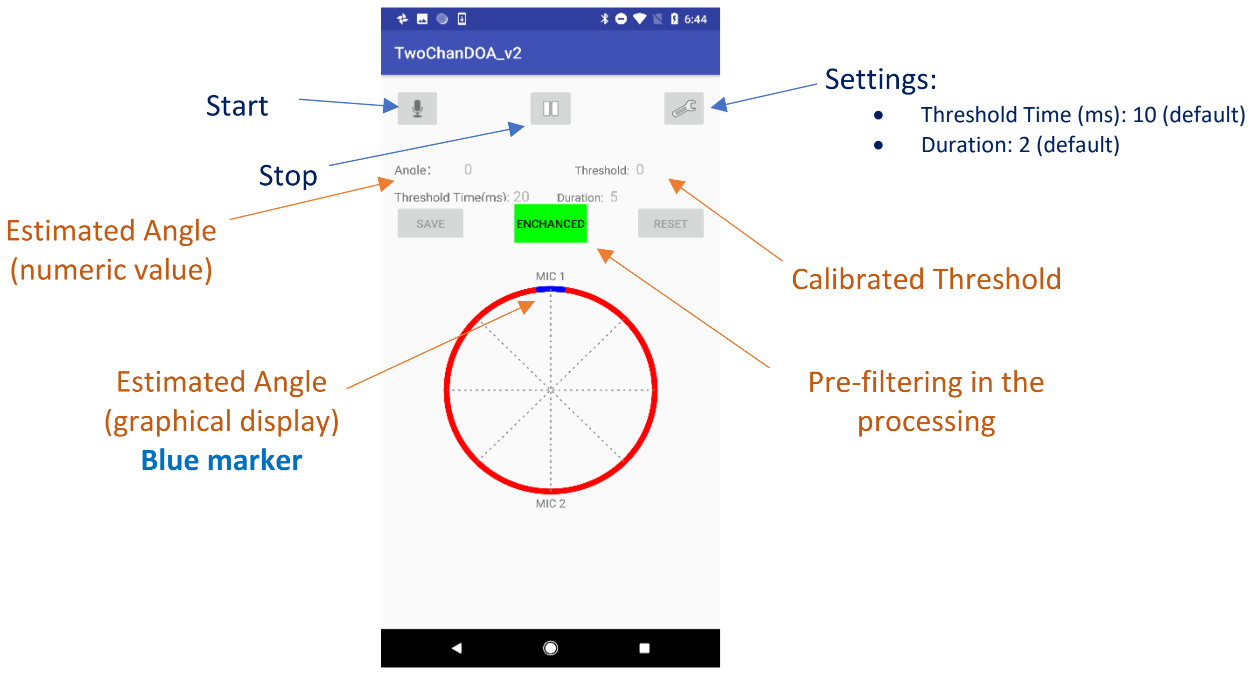


- Firstly, we use Graph class to read the model from the directory. Then, we utilize TensorFlowInferenceInterfence class to inferring. We employ "feed", "run", and "fetch" methods to run inference. The result of the "fetch" method is probabilities of each class in our case. Hence, the max of it should be found in order to estimate most probable class.

5. TWO MIC DIRECTION OF ARRIVAL (DOA) ESTIMATION APPLICATION

5.1 DOA ESTIMATION APPLICATION 1

The algorithm is developed for providing the hearing-impaired user with an estimate of the speaker direction on a graphical user interface (GUI). The application is trained to perform in noisy conditions as well. See Figure 8 for a screenshot of the Android application.


6:44
TwoChanDOA_v2
Start
Stop
Settings:
• Threshold Time (ms): 10 (default)
• Duration: 2 (default)
Estimated Angle (numeric value)
Calibrated Threshold
SAVE
ENCHANCED
RESET
MIC 1
Estimated Angle (graphical display)
Blue marker
Pre-filtering in the processing
MIC 2

Angle: 0
Threshold: 0
Threshold Time(ms): 20
Duration: 5

Figure 6

Following are the key things to keep in mind while using the application:

- Before starting the application, please make sure the Smartphone is placed on an elevated surface (so that the microphones are not covered). The GUI should still be fully visible to the user.
- A new button, “Enhanced”, is added with new version. When the button is red, the app will operate like version 1. If the button is green, pre-filtering will be enabled and the app will estimate better than version 1.

- Press the 'Start' button to start the application. The algorithm assumes that the first few frames are silence/noise only. Threshold Time specifies this duration (which can be modified using the 'Settings' button).
- After the initial few frames, the 'Calibrated Threshold' display will stop at a fixed value. Now the application is ready and you may start speaking.
- The application will receive the microphone signals and indicate the direction of the speaker using numeric values as well as using graphical marker.
- The behavior of the application can be modified using 'Settings' button. Increasing the 'Threshold time' will allow the algorithm to be more noise-robust. Increasing the 'Duration' will make the DOA estimation marker more stable and change less rapidly.
- In presence of speech, the blue marker will continuously follow the speaker in all directions. In absence of speech, the blue maker will point to the last estimated speaker location, even with a high-energy noise source present in the room (such as Vacuum cleaner or coffee machine).

To learn more about the application, please refer to our video demos on http://www.utdallas.edu/ssprl/hearing-aid-project/.

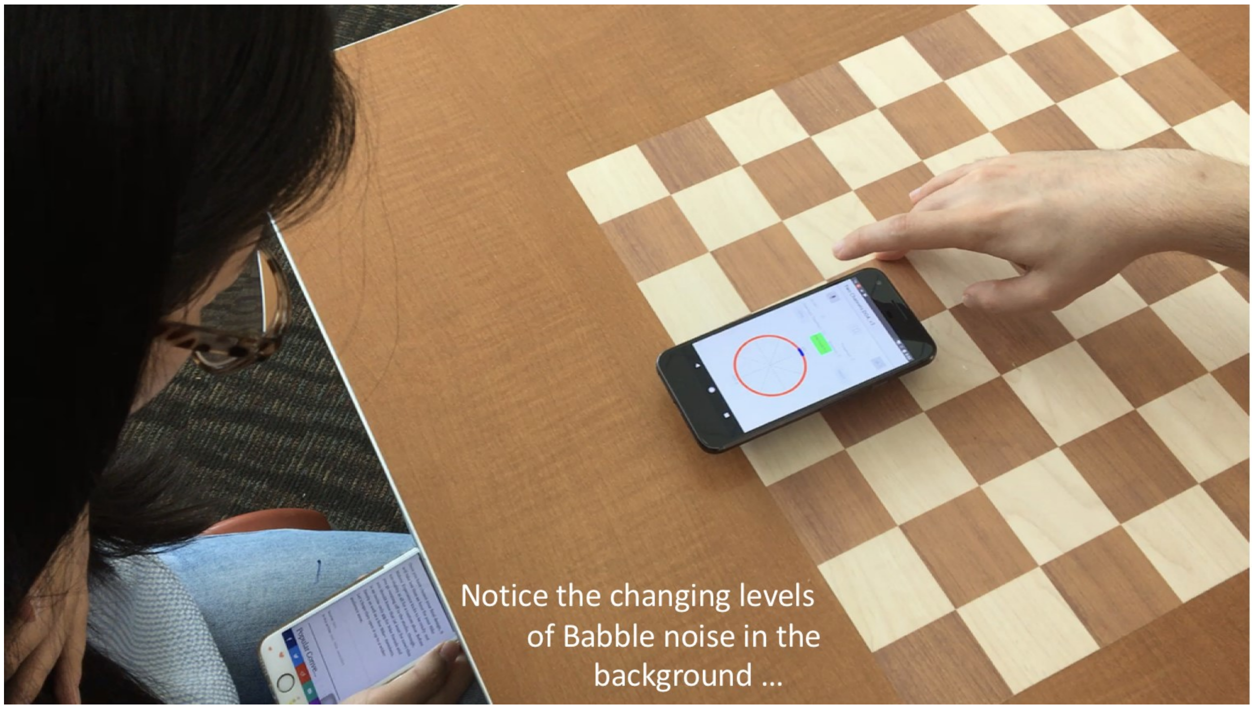
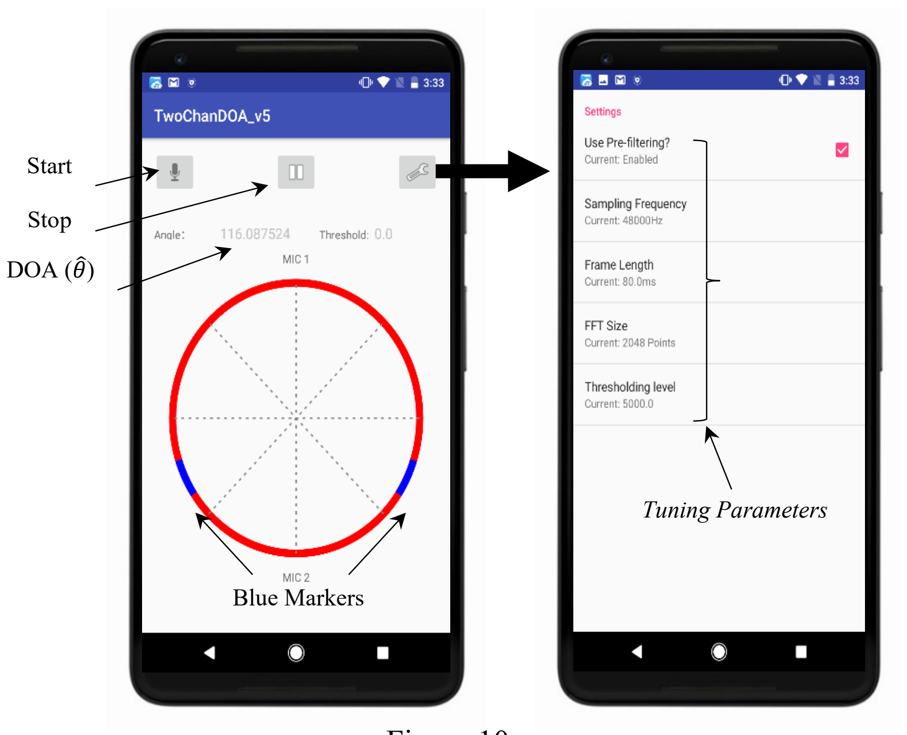


Figure 9 Video screenshots for Demo videos

5.2 DOA ESTIMATION APPLICATION 2

The algorithm is developed for providing the hearing-impaired user with an estimate of the speaker direction on a graphical user interface (GUI). The application is trained to perform in noisy conditions as well. See Figure 10 for a screenshot of the Android application.



Angle: 116.087524 Threshold: 0.0

Settings

Use Pre-filtering? Current: Enabled

Sampling Frequency Current: 48000Hz

Frame Length Current: 80.0ms

FFT Size Current: 2048 Points

Thresholding level Current: 5000.0

Figure 10

Following are the key things to keep in mind while using the application:

- Before starting the application, please make sure the Smartphone is placed on an elevated surface (so that the microphones are not covered). The GUI should still be fully visible to the user.
- A new screen is designed for Settings. Preferences can be tuned used this screen before beginning the application.
- Press the ‘Start’ button to start the application. You can start speaking immediately, you don’t need wait a few frames like previous app.
- The application will receive the microphone signals and indicate the direction of the speaker using numeric values as well as using graphical marker.

- The behavior of the application can be modified using ‘Settings’ button. Increasing the ‘Thresholding Level’ will make the DOA estimation marker more stable and change less rapidly.
- In presence of speech, the blue marker will continuously follow the speaker in all directions. In absence of speech, the blue maker will point to the last estimated speaker location, even with a high-energy noise source present in the room (such as Vacuum cleaner or coffee machine).

To learn more about the application, please refer to our video demos on http://www.utdallas.edu/ssprl/hearing-aid-project/.

Figure 11 Video screenshots for Demo videos

5.3 DOA ESTIMATION APPLICATION 3

The algorithm is developed for providing the hearing-impaired user with an estimate of the speaker direction on a graphical user interface (GUI). This application is convolutional neural network (CNN) based. The CNN model is trained to perform in noisy conditions as well. See Figure 10 for a screenshot of the Android application.

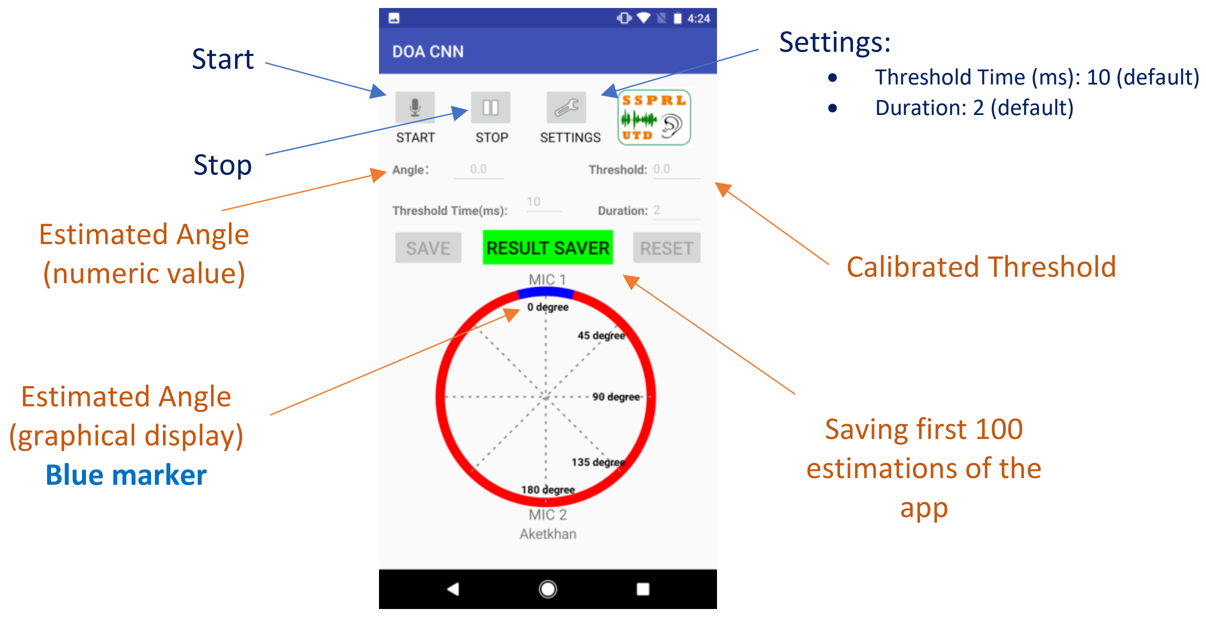


Figure 12

Following are the key things to keep in mind while using the application:

- Before starting the application, please make sure the Smartphone is placed on an elevated surface (so that the microphones are not covered). The GUI should still be fully visible to the user.
- Press the 'Start' button to start the application. You can start speaking immediately, you don't need wait a few frames like previous app.
- The application will receive the microphone signals and indicate the direction of the speaker using numeric values as well as using graphical marker.
- The behavior of the application can be modified using 'Settings' button. Increasing the 'Thresholding Level' will make the DOA estimation marker more stable and change less rapidly.

- In presence of speech, the blue marker will continuously follow the speaker in all directions. In absence of speech, the blue maker will point to the last estimated speaker location, even with a high-energy noise source present in the room (such as Vacuum cleaner or coffee machine).
- You can save the result by making "Result Saver" button green. When you touch the "Result Saver" button, it will ask the speaker angle according to the phone. After you input the angle of speaker, it will store 100 estimations of the app and calculate the accuracy and root mean square error (RMSE) of the app.

To learn more about the application, please refer to our video demos on http://www.utdallas.edu/ssprl/hearing-aid-project/.

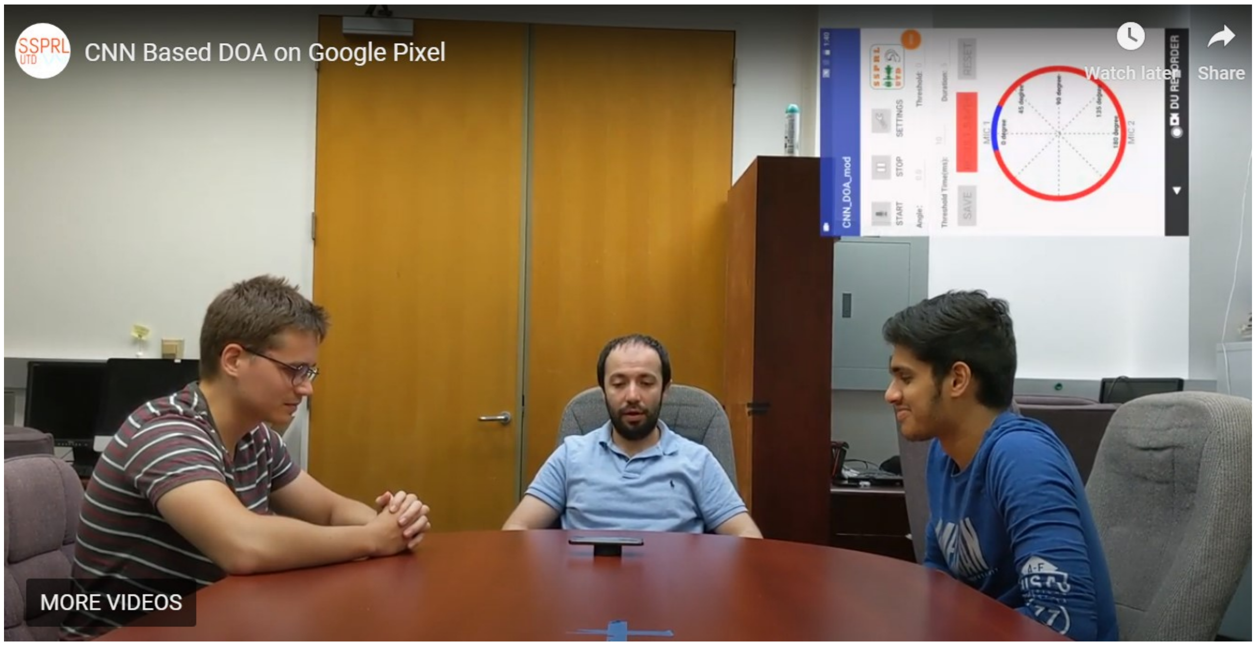


Figure 13 Video screenshots for DOA estimation application version 3

6. REFERENCES

[1] Abdullah Küçük, Yiya Hao, Anshuman Ganguly, Issa Panahi, "Stereo I/O Framework for Audio Signal Processing on Android Platforms", Proceedings of Meetings on Acoustics, Acoustic Society of America (ASA 2018), Minneapolis, MN, May 2018.

[2] https://developer.android.com/reference/android/media/AudioRecord.html

[3] https://developer.android.com/ndk/guides/audio/sampling-audio.html